

Laboratorio 1: Gestión de Procesos

Estados de Procesos

Para observar los distintos estados de un proceso en Windows, utilicé un programa en Python que simula las transiciones típicas: Nuevo, Listo, Ejecutando, Bloqueado y Terminado.

Al ejecutar el script `estados_proceso.py`, se capturaron los siguientes estados y tiempos de transición:

```
C:\Users\Eduardo>cd C:\Users\Eduardo\Desktop\Laboratorios_SO_Windows_Eduardo\Laboratorio_1_Gestion_de_Procesos
C:\Users\Eduardo\Desktop\Laboratorios_SO_Windows_Eduardo\Laboratorio_1_Gestion_de_Procesos>python estados_proceso.py
Estado: Nuevo
→ Tiempo desde inicio: 1.00 segundos
Estado: Listo
→ Tiempo desde Nuevo a Listo: 1.01 segundos
Estado: Ejecutando
Procesando...
Procesando...
Procesando...
→ Tiempo desde Listo a Ejecutando: 3.01 segundos
Estado: Bloqueado (esperando recurso)
→ Tiempo desde Ejecutando a Bloqueado: 2.00 segundos
Estado: Terminado
→ Tiempo total desde inicio: 7.02 segundos
C:\Users\Eduardo\Desktop\Laboratorios_SO_Windows_Eduardo\Laboratorio_1_Gestion_de_Procesos>
```

Estas transiciones reflejan los pasos comunes que atraviesa un proceso en el sistema operativo: desde su creación (Nuevo), hasta estar listo para ejecutarse (Listo), pasar a la ejecución efectiva (Ejecutando), detenerse temporalmente mientras espera un recurso (Bloqueado) y finalmente su finalización (Terminado).

Scheduling del Sistema Operativo

Para estudiar cómo el sistema operativo distribuye el tiempo de CPU, se ejecutaron simultáneamente cinco programas que demandan gran cantidad de recursos de procesamiento.

Durante la ejecución, se monitoreó el uso de CPU en el Administrador de tareas de Windows. Se observaron los siguientes porcentajes aproximados de uso de CPU para cada proceso:

- Cada uno de los cuatro programas consumía alrededor del 17.4% de la CPU y uno 17,8%.
- Uno de los procesos llegó a un pico del 19.0%.

Este comportamiento refleja una distribución relativamente equitativa del tiempo de CPU entre los procesos, similar a algoritmos de planificación de tipo Round Robin.

Esto indica que el sistema operativo priorizó los primeros cuatro procesos, dejando al quinto proceso sin recursos para ejecutarse. Este comportamiento se asemeja al algoritmo de planificación FIFO (First In, First Out), donde los primeros procesos en llegar se ejecutan hasta finalizar, mientras que los que llegan después deben esperar. En este caso, el proceso 5 no llegó a ejecutarse, lo cual refleja una falta de equidad en la distribución de recursos.

Si se hubiera utilizado un algoritmo de planificación Round Robin, el comportamiento habría sido distinto. Este tipo de planificación asigna un pequeño intervalo de tiempo a cada proceso de forma rotativa, permitiendo que todos avancen gradualmente. Bajo este esquema, todos los procesos, incluido el proceso 5, habrían tenido oportunidad de ejecutarse, aunque fuera parcialmente.

Administrador de tareas

Archivo Opciones Vista

Procesos		Rendimiento	Historial de aplicaciones	Inicio	Usuarios	Detalles	Servicios
		^					
Nombre		Estado	100% CPU	80% Memoria	1% Disco	0% Red	C
Aplicaciones (7)							
>	Administrador de tareas		6,6%	23,3 MB	0,1 MB/s	0 Mbps	
>	Microsoft Edge (9)		0,5%	85,7 MB	0 MB/s	0 Mbps	
>	Procesador de comandos de Wi...		19,0%	13,3 MB	0 MB/s	0 Mbps	
>	Procesador de comandos de Wi...		17,8%	13,3 MB	0 MB/s	0 Mbps	
>	Procesador de comandos de Wi...		17,4%	13,3 MB	0 MB/s	0 Mbps	
>	Procesador de comandos de Wi...		17,4%	13,3 MB	0 MB/s	0 Mbps	
>	Procesador de comandos de Wi...		17,4%	13,3 MB	0 MB/s	0 Mbps	

Creación y Observación de Deadlock

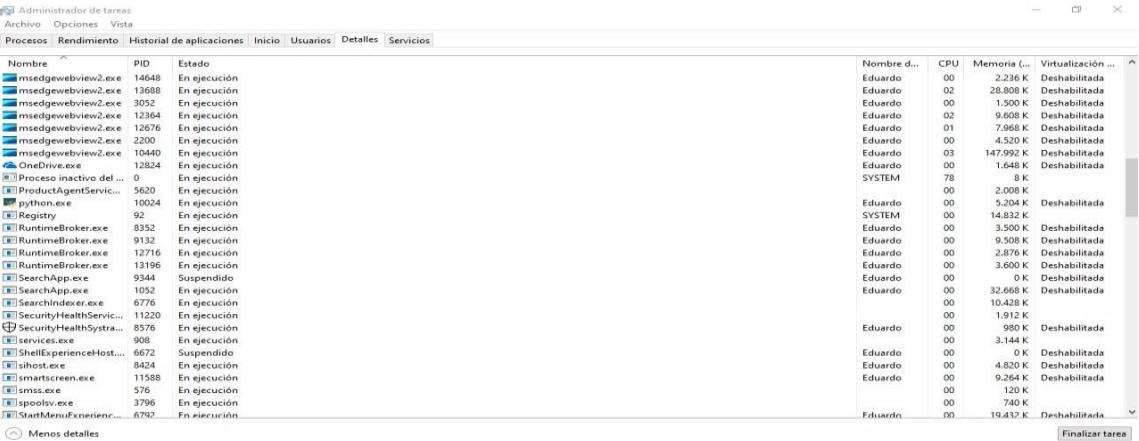
Para demostrar un deadlock, se implementó un programa en Python con dos hilos que intentan acceder a dos recursos (locks) en orden inverso, generando un bloqueo mutuo.

Durante la ejecución, el programa queda bloqueado esperando indefinidamente la liberación de recursos, sin avanzar ni terminar.

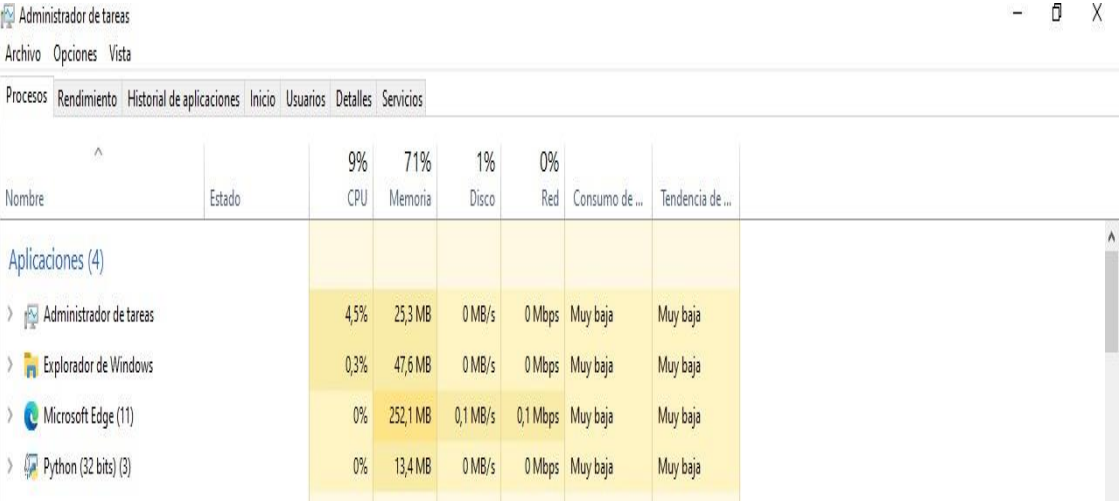
En el Administrador de tareas, se puede observar el proceso de Python activo, pero sin un consumo significativo de CPU.

```
C:\Users\Eduardo>cd Desktop\Laboratorios_SO_Windows_Eduardo\Laboratorio_1_Gestion_de_Procesos

C:\Users\Eduardo\Desktop\Laboratorios_SO_Windows_Eduardo\Laboratorio_1_Gestion_de_Procesos>python deadlock.py
Hilo 1: intentando acceder a Recurso A
Hilo 1: accedió a Recurso A
Hilo 2: intentando acceder a Recurso B
Hilo 2: accedió a Recurso B
Hilo 2: intentando acceder a Recurso A
Hilo 1: intentando acceder a Recurso B
```



Nombre	PID	Estado	Nombre d...	CPU	Memoria (...)	Virtualización ...
msedgeview2.exe	14648	En ejecución	Eduardo	00	2,236 K	Deshabilitada
msedgeview2.exe	13688	En ejecución	Eduardo	02	28,808 K	Deshabilitada
msedgeview2.exe	3052	En ejecución	Eduardo	00	1,500 K	Deshabilitada
msedgeview2.exe	12364	En ejecución	Eduardo	02	9,608 K	Deshabilitada
msedgeview2.exe	12676	En ejecución	Eduardo	01	7,968 K	Deshabilitada
msedgeview2.exe	2200	En ejecución	Eduardo	00	4,520 K	Deshabilitada
msedgeview2.exe	10440	En ejecución	Eduardo	03	147,992 K	Deshabilitada
OneDrive.exe	12824	En ejecución	Eduardo	00	1,648 K	Deshabilitada
Proceso inactivo del ...	0	En ejecución	SYSTEM	78	8 K	Deshabilitada
ProductAgentServic...	5620	En ejecución	Eduardo	00	2,008 K	Deshabilitada
python.exe	10024	En ejecución	Eduardo	00	5,204 K	Deshabilitada
Registry	92	En ejecución	SYSTEM	00	14,832 K	Deshabilitada
RuntimeBroker.exe	8352	En ejecución	Eduardo	00	3,500 K	Deshabilitada
RuntimeBroker.exe	9132	En ejecución	Eduardo	00	9,508 K	Deshabilitada
RuntimeBroker.exe	12716	En ejecución	Eduardo	00	2,876 K	Deshabilitada
RuntimeBroker.exe	13196	En ejecución	Eduardo	00	3,600 K	Deshabilitada
SearchApp.exe	9344	Suspendido	Eduardo	00	0 K	Deshabilitada
SearchApp.exe	1052	En ejecución	Eduardo	00	32,668 K	Deshabilitada
SearchIndexer.exe	6776	En ejecución	Eduardo	00	10,428 K	Deshabilitada
SecurityHealthServic...	11220	En ejecución	Eduardo	00	1,912 K	Deshabilitada
SecurityHealthSyste...	8576	En ejecución	Eduardo	00	980 K	Deshabilitada
services.exe	908	En ejecución	Eduardo	00	3,144 K	Deshabilitada
ShellExperienceHost...	6672	Suspendido	Eduardo	00	0 K	Deshabilitada
sihost.exe	8424	En ejecución	Eduardo	00	4,820 K	Deshabilitada
smartscreen.exe	11588	En ejecución	Eduardo	00	9,264 K	Deshabilitada
smss.exe	576	En ejecución	Eduardo	00	120 K	Deshabilitada
spoolsv.exe	3796	En ejecución	Eduardo	00	740 K	Deshabilitada
SearchMenuExperienr...	6792	En ejecución	Eduardo	00	19,432 K	Deshabilitada



Nombre	Estado	9%	71%	1%	0%	Consumo de ...	Tendencia de ...
		CPU	Memoria	Disco	Red		
Aplicaciones (4)							
Administrador de tareas		4,5%	25,3 MB	0 MB/s	0 Mbps	Muy baja	Muy baja
Explorador de Windows		0,3%	47,6 MB	0 MB/s	0 Mbps	Muy baja	Muy baja
Microsoft Edge (11)		0%	252,1 MB	0,1 MB/s	0,1 Mbps	Muy baja	Muy baja
Python (32 bits) (3)		0%	13,4 MB	0 MB/s	0 Mbps	Muy baja	Muy baja

Se implementó un mecanismo en el código para detectar el deadlock mediante un timeout, que reporta la situación en la consola antes de terminar.

Conclusión

En este laboratorio se logró comprender el ciclo de vida de un proceso dentro del sistema operativo, observando sus diferentes estados y los tiempos de transición entre ellos. Esto permitió visualizar cómo el sistema gestiona y controla la ejecución de procesos en un entorno real.

La ejecución simultánea de varios programas que demandan recursos de CPU evidenció que el sistema operativo distribuye el tiempo de procesamiento de manera equitativa, evitando que un solo proceso monopolice la CPU. Este comportamiento se asemeja a algoritmos de planificación como Round Robin, que buscan maximizar la eficiencia y la justicia en la asignación de recursos. Finalmente, la creación deliberada de un deadlock mostró cómo dos procesos pueden quedar bloqueados indefinidamente al esperar recursos en orden inverso, lo cual detiene la ejecución normal del sistema. La detección y análisis de este estado resaltan la importancia de implementar mecanismos para evitar o resolver deadlocks en sistemas reales, asegurando así la estabilidad y disponibilidad del sistema operativo.